3-D Object Modeling and Recognition for Telerobotic Manipulation

Andrew Johnson Patrick Leger Regis Hoffman Martial Hebert James Osborn

The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

This paper describes a system that semi-automatically builds a virtual world for remote operations by constructing 3-D models of a robot's work environment. With a minimum of human interaction, planar and quadric surface representations of objects typically found in man-made facilities are generated from laser rangefinder data. The surface representations are used to recognize complex models of objects in the scene. These object models are incorporated into a larger world model that can be viewed and analyzed by the operator, accessed by motion planning and robot safeguarding algorithms, and ultimately used by the operator to command the robot through graphical programming and other high level constructs. Limited operator interaction, combined with assumptions about the robots task environment, make the problem of modeling and recognizing objects tractable and yields a solution that can be readily incorporated into many telerobotic control schemes.

1 Introduction

Numerous teleoperation tasks require manipulators to move objects and position end effectors; typically, the operator utilizes 2-D data from a suite of video cameras to plan and execute tasks. The reliance on cameras limits the effectiveness of such teleoperated systems because video images provide only 2-D information. For many complex tasks, 3-D information is vital.

Previous work uses a variety of sensors and representations to construct models of interior workspaces. Christensen [1] describes a supervised teleoperated system with a world model that includes a priori knowledge, the robot configuration, and information gathered by the robot's sensors. The world model allows the operator to preview the operation, have the computer automatically plan an end-effector trajectory, and view the task from many different viewing positions. Trivedi [2] reports on another model-based system using range sensors; this system allows testing of robot plans in simulation. Azarbayejani [3] has a system to semiautomatically construct CAD models from uncalibrated video images. Thayer [4] computes an object's location and orientation using stereo vision; the operator performs the stereo matching of some points on the object and the system computes the pose of the object.

These and other modeling systems share common characteristics that justify their development:

1. They serve as a graphical aid to the operator by allowing viewing the robot and its workspace from many more angles than are possible with on-board cameras.
2. They store knowledge of the world in the form of a mathematical model; this capability is essential for automatic planning agents.
3. They augment uncertain or unrecorded a priori information with up-to-date in situ data.

The system we describe generates 3-D models of man-made environments from laser rangefinder data to produce a virtual 3-D world for the human operator. This capability is critical for future efforts in which teleoperated systems evolve into telerobotic and autonomous systems.

2 Telerobotics and 3-D object modeling

The Department of Energy has declared that robots and remote systems will play crucial roles in future decontamination and decommissioning (D&D) of nuclear weapons facilities [5]. Mobile worksystems will be used in the near term for selective equipment removal, in which some part of an apparatus is extricated while minimally disturbing the surrounding objects. An example of a mobile

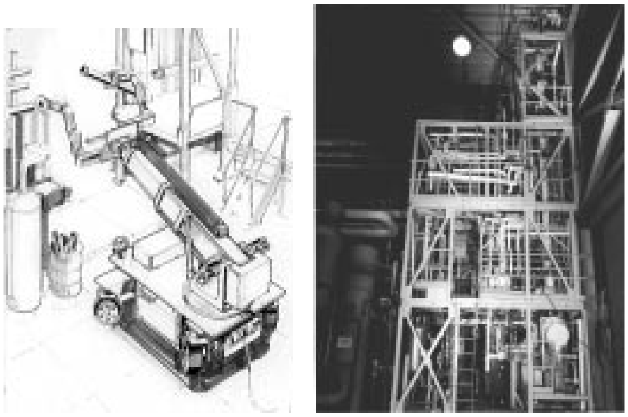

Figure 1: The Rosie Mobile Worksystem removing a section of pipe (left). The IODOX facility, a DOE test site for selective equipment removal (right).

worksystem is “Rosie,” an advanced prototype for testing, evaluating and demonstrating robotic selective equipment removal[6]. Rosie includes a locomotor, a heavy manipulator, an operator control center, and a control system for robot operation. An key component of the operator control center is the 3-D object modeling and recognition system. Figure 1 shows the worksystem cutting and removing a section of pipe using the DOE dual-arm work module, Rosie’s principal payload. A facility representative of those in which Rosie and other worksystems will operate is shown in Figure 1. The near term goal of the work described herein is to enable dismantlement of such facilities using telerobotic control schemes that provide substantial productivity gains relative to baseline teleoperation. Accurate 3-D object modeling is the essential foundation for telerobotic planning and motion control.

Our modeling system is a combination of sensors, modeling and analysis software, and an operator interface that creates 3-D models of indoor man-made environments as they are discovered. The interactive system described performs the following functions:

1.Acquire 3-D scene data in the form of range images.
2. Select a region of interest in each range image.
3. Generate a surface mesh within the region of interest.
4. Segment mesh into planar and quadric surfaces patches.
5. Match surface patches to object models for object recognition.
6. Incorporate object models into robot’s world model.

A block diagram of the above operations and data flow is shown in Figure 2. Automatic actions are shown on the left while operator interactions are given on the right. The remainder of this paper details these steps and concludes with a detailed description of the propagation of errors through the system

3 Data acquisition

The sensor used in our experiments is a 3-D scanning laser rangefinder manufactured by Perceptron, Inc. It acquires 256 x 256 pixel range and intensity images over a vertical and horizontal field of view of 60 degrees at a frame rate of 2 Hz. The scanner’s range is 2 to 40 meters and its range precision is 5-7 cm [7]; the sensor that will be used in the final system will have much better accuracy and precision. To map a facility, the scanner is remotely positioned by a mobile worksystem and commanded by the human operator to acquire a sequence of images. Linear fitting of image data at known range values is used to calibrate the sensor for range scale and offset due to temperature drift of the sensor which improves the range accuracy to the centimeter range. The range and intensity images acquired are displayed on the operator’s console. Figure 3 shows a sample image and reflectance image pair taken at our experimental test site.

To focus the system on specific objects in the scene to be analyzed, the operator selects a rectangular *region of interest* in the image. This limits the amount of data to that which is important to recognizing the objects of interest. Figure 3 shows a region of interest (interior of the white rectangle in the range image) selected by an operator interested in determining the position of the t-joint and diagonal pipe imaged by the sensor.

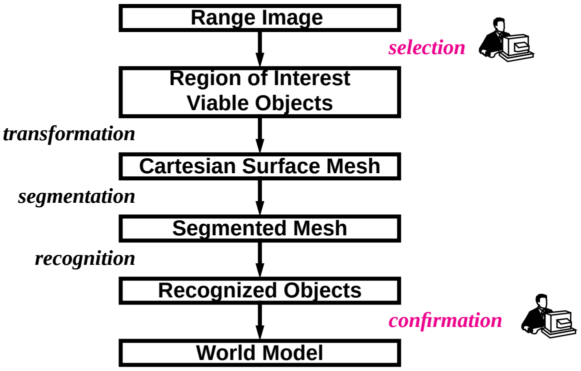

Figure 2: Modeling system block diagram.

Once the region of interest is determined a spatial smoothing filter is applied to eliminate outliers while preserving range discontinuities. The resulting range image is then converted from spherical sensor coordinates (ρ,θ,ϕ) to cartesian world coordinates (x,y,z). To group points into surface patches, a surface mesh is created that establishes the local connectivity of the points in space based on the connectivity of pixels in the range image. Nearest neighboring range points are connected if the distance between them is less than a specified threshold. (this preserves range step discontinuities when converting from the sensor to world coordinates). The resulting set of connected points constitutes a surface mesh in cartesian space which is used in all subsequent processing stages.

Figure 4 shows the surface mesh generated from the region of interest selected in Figure 3. For presentation purposes, the two pipes that form a T-joint are labelled “T1” and “T2”; the pipe beneath (and occluded by) the T-joint is labelled “P” and the two prominent flat regions are labelled “F1” and “F2”.

4 Object representation

We have chosen a surface based object representation (as opposed to volume based) because we are building world models which will be used to plan complex tasks. A volume based representation is not sufficient for planning tasks such as grasping an object, applying a surface coating, or obtaining a sample. The robot workspace contains large numbers of man-made objects with fairly simple geometries but with complex arrangements and numerous interconnections (e.g., Figure 1). Curved objects, such as pipes and vessels, can be concisely modeled with *quadric* surfaces - second order surface descriptions having ten parameters, while flat objects, such as walls, floors and support structures can be modeled with planar surfaces. Quadric and planar surfaces are suitable

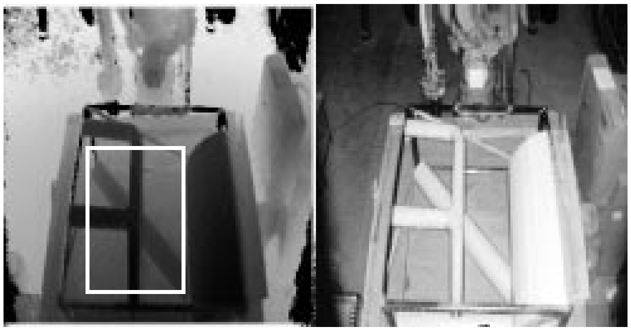

Figure 3: Range image and selected region of interest (left) and reflectance image (right) of experimental testbed.

descriptions for 3-D modeling of man-made interiors because their parametric representations do not vary when surfaces are partially occluded or the sensing viewpoint changes. Furthermore, the position and orientation of surface patches in the scene can be easily computed from their corresponding parameters.

In the remainder of this paper the following notation will be used. A 3-D point *x* lies on a planar surface with surface normal *n* and cartesian distance *d* from the origin of the world coordinate frame if it satisfies:

$$n^t x + d = 0 \qquad (1)$$

A 3-D point *x* lying on a quadric surface satisfies:

$$x^t A x + B^t x + d = 0 \qquad (2)$$

where *A* is a 3x3 matrix describing the curvature of the surface, *B* is a 3x1 matrix describing the position of the quadric with respect to the world coordinate system origin, and *d* is a scalar related to the size of the quadric.

5 Segmentation

Our segmentation method is based on surface meshes that group range points into surfaces and is independent of the means by which the range data is collected as long as the data is transformable into a mesh representation. This is attractive as it allows the use of the range sensing modality (structured light, depth from focus, laser rangefinding, stereo vision, etc.) that is best suited to the task environment. Though the results described herein are based on meshes obtained from single range images, the technique can be applied to data obtained from multiple views or different sensors. Substantial performance improvements through reducing the effects of occlusions, isolated range errors and increasing the sensor field of view are thus possible. A general surface mesh segmentation technique is essential to realizing these benefits.

The similarity measure for segmentation in our algorithm is the proximity of the points to planar or quadric surfaces in the scene. The segmentation groups points into *surface patches* which are described by a set of points and the best fit parameters of the surface fit to the points. This parametric description of surface patches makes the task of object recognition more efficient.

Our algorithm, which is based on the algorithm for segmenting range images presented by Faugeras and Hebert [8], is presented in detail in [9]. It is a generalization of their technique because it performs the segmentation using the arbitrary connectivity of a surface mesh. Furthermore, the similarity measure that defines when to merge two regions can be easily changed to measure any property of the surface.

The first stage in the algorithm segments a mesh into planar surface patches using global region growing. Initially, each point in the mesh is considered a region; the two regions with the minimum merge error given by

$$\sum_{i=1}^{N}\left(n^t x_i + d\right)^2 \qquad (3)$$

where *i* is over point in both regions are merged. This merging continues until the total merge error exceeds some user defined threshold. The result is a set of planar surface patches on the surface mesh.

The algorithm for creating quadric surface patches is very similar to that used to generate planar surface patches. First, quadric patches are generated from each sufficiently large planar patch by calculating the quadric parameters of each patch. Then the two quadric patches with the minimum merge error given by

$$\sum_{i=1}^{N}\left(x_i^t A x_i + B^t x_i + d\right)^2 \qquad (4)$$

are merged. This merging continues until the total merge error exceeds some user defined threshold.

The result of the segmentation is a set of quadric and planar surface patches. Figure 5 shows the result of applying the segmentation to the mesh given in Figure 4. The points that lie on the curved surfaces in the scene have been merged into single curved quadric surfaces (T1,T2,P) while points lying on flat surfaces have been merged into planar surfaces (F1,F2), not quadric surfaces. Points that lie on different pipes of the T-joint have not been merged because they belong to the distinct quadrics. The two patches that lie on P have not been merged due to the occlusion by the pipe T-joint in the foreground.

The increased utility of performing the segmentation on a surface mesh comes at the cost of added representational complexity of the data in the algorithm.

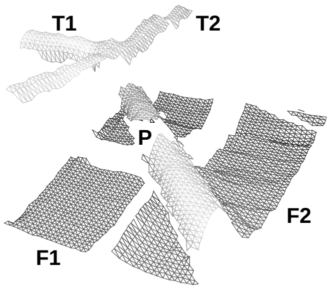

Figure 4: Result of constructing a surface mesh from the region of interest shown in Figure 3.

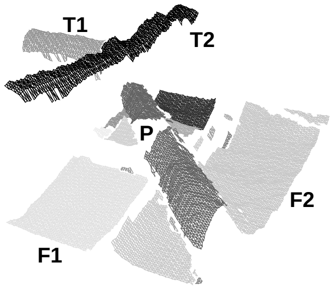

Figure 5: An oblique view of the final planar and quadric surface patches. Prominent patches are identified with labels given in Figure 4.

The surface mesh and the region adjacency graphs must be represented with undirected parameterized graphs to describe their full 3-D connectivity. Segmentation techniques based on images can use simple functions to access pixels, while lookup functions on graphs are complicated to ensure efficiency.

6 Object recognition

The purpose of the mesh segmentation is to group points in the scene into higher level constructs that can be used more efficiently by the object recognition system. The basic purpose of the object recognition system is to locate models of objects in the scene. To make object recognition tractable, the system relies on the nature of the structured environment to limit the number of possible objects in the scene, and on operator interaction to verify the location of the objects found by the computer. First, the operator selects one or more object types he recognizes in the region of interest from a menu of objects in a library. Then, the object recognition module searches for these models in the scene. When a model is found, it is presented to the operator for verification before being inserted into the world model at the pose computed by the object recognition module. The human interaction is limited to selecting the objects to look for and verifying their location once they are found. The object recognition methodology presented, mentioned by Besl in [10], relies on aligning principal directions of the scene patches with principal directions of model patches to compute a transformation.

6.1 Matches between quadrics

Our object recognition system determines the transformation between a model surface patch represented by a set of surface parameters and a scene surface patch represented by a set of surface parameters and a set of points that are grouped by the patch.

In the first case, we will consider a match between a model quadric patch, denoted with subscript *m*, and a scene quadric patch, denoted with subscript *s*. Suppose that the scene patch corresponds to a rotated and translated version of the model patch. Then there exists some rotation *R* and translation *T* such that

$$(Rx+T)^t A_m (Rx+T) + B_m^t (Rx+T) + d_m = x^t A_s x + B_s^t x + d_s \qquad (5)$$

from which can be derived

$$A_m' = R^t A_m R = A_s \qquad (6)$$

$$B^t_m{}' = 2T^t A_m R + B^t_m R = B^t_s \qquad (7)$$

$$d_m' = T^t A_m T + B_m T + d_m = d_s \qquad (8)$$

The eigenvectors of A_m correspond to the principal directions of the model quadric and the eigenvectors of A_s correspond to the principal directions of the scene quadric. If the two patches match each other then Equation 6 states that the *R* must rotate the principal directions of the scene patch onto the principal directions of the model patch. It is unknown which principal directions in the model correspond to which principal directions in the scene, so there exists six possible rotations that will rotate the model principal directions onto the scene principal directions. These rotations are generated by matching eigenvectors of the model patch to eigenvectors of the scene patch. The eigenvectors of the scene and the model have arbitrary signs (i.e., they can point in either direction along the principle axes of the quadric), so for each of the six transformations that align the principle axes of the quadrics there exist four different rotations. Hence, there exists twenty four possible rotations for the transformation of model to scene. Once the twenty four matches between model and scene eigenvectors have been enumerated, the corresponding twenty four rotation matrices are generated using the quaternion representation presented in [8].

For each of these twenty four rotations, Equation 7 is solved for *T*, resulting in twenty four (*R*,*T*) pairs. For each of the (*R*,*T*) pairs the new quadric parameters A_m', B_m' and d_m' of the rotated model patch are calculated from the left hand sides of Equation 6, 9 and 10. The best (*R*,*T*) pair for the match is then selected as the one that minimizes the match error measure

$$E = \frac{1}{N}\sum_{i=1}^{N}\left(x_i^t A_m' x_i + B_m'^t x_i + d_m'\right)^2 \qquad (9)$$

for all of the points in the scene patch. In the ideal case, where scene points lie on a rotated and translated version of model patch, the best (*R*,*T*) pair will set A_m', B_m' and d_m' equal to A_s, B_s and d_s respectively, and Equation 9 will equal zero. This match error measure will reward transformations that transform the model patch parameters to similar values as the scene patch parameters and will penalize ones that do not. Note, however, that the match error measure does not give the Euclidean distance between the scene points and the rotated model patch.

Each possible rotation results from pairing axes along the principal directions of the scene patch with axes along the principal directions of the model patch. A *match* between a scene patch and a model patch will consist of the pairings of these axes, the transformation that maps the model patch to the scene patch and the match error. In the case of a non-singular model quadric, the match contains three axis pairs.

Special cases occur when the principal directions of a quadric are ambiguous due to an axis of symmetry. These special cases are dealt with thoroughly, but limited paper space precludes presenting the details.

6.2 Matches between planes

The procedure for finding the best transformation between a model planar patch and a scene planar patch is similar to that presented for quadric patches. Suppose that the planar scene patch corresponds to a rotated and translated version of the planar model patch. Then there exists some rotation *R* and translation *T* such that

$$n_m^t (Rx + T) + d_m = n_s^t x + d_s \tag{10}$$

from which can be derived

$$n_m' = n_m^t R = n_s \tag{11}$$

$$d_m' = n_m^t T + d_m = d_s \tag{12}$$

Equation 11 requires the rotation *R* to rotate the unit scene normal to the unit model normal. This rotation is the rotation about $n_m \, x \, n_s$ by an angle $\cos^{-1}(n^t_m \, n_s)$. The appropriate translation *T* can be determined by fixing two coordinates of *T* and solving for the third using Equation 12. The ambiguity in the scene surface normal is eliminated by forcing the inner product between the scene surface normal and the ray from the sensor to be positive. Similar to the case of matching quadrics, the rotated model plane parameters n_m and d_m are calculated from the left hand sides of Equation 11 and 13 and the match error for a model and scene planar patch is

$$E = \frac{1}{N} \sum_{i=1}^{N} \left(n_m^{t\prime} x_i + d_m' \right)^2 \tag{13}$$

A match between a model planar patch and a scene patch will have one axis match, the normals of the scene and model patches.

Any model planar patch can be transformed exactly to any scene planar patch, so Equation 13 will always be zero when only two planar patches are matched. However, complex objects will have more than one plane making an exact transformation less common when more than one model plane is matched to the scene.

6.3 Searching for complex objects

Single quadric or planar surfaces are sufficient for modeling simple objects like walls and pipes but are not sufficient for modeling more complex objects. Fortunately, models of complex objects which are prevalent in man-made environments, like pipe joints, I-beams and holding vessels, can be created by grouping together multiple planar and quadric surfaces. This representation conveys only the surfaces that make up a model and not does not represent the boundaries between surfaces in a model, so explicit connectivity between surfaces is not represented. However, for locating the pose of objects in the scene, this representation is sufficient. The size of the model is explicit in the parameters of the surface that make up the model, so different models must be created for similar objects of varying size. Searching for models made from multiple surfaces in the scene relies on the techniques presented for finding a match between single surfaces as presented in the previous two sections.

The model in the scene is found by matching model patches to scene patches and computing the transformation of the model from the match. First, the model patch with the highest discernability (a measure based on the size of the patch and its ability to fix the pose of the patch in the scene) M_1 is matched to each scene patch and the transformations between scene and model patches are computed. Matches with errors (as given by Equation 9 and Equation 13) that are below a pre-determined match error threshold are inserted onto a list of feasible matches ***F*** because they are good matches between the parameters of the model patch and the points grouped by the scene patch. All of the other matches are eliminated.

Next, the model patch M_2 with the second highest discernability is matched to every patch in the scene and the transformations and the match errors between scene patches and M_2 are computed. Of these matches, the ones with errors below the match error threshold are kept in a temporary list of feasible matches $\boldsymbol{F_2}$. Then each match in $\boldsymbol{F_2}$ is combined with each match in ***F*** by concatenating the lists of axis pairs of each match. The rotations, translations and match errors of the combined matches are computed as explained below. The combined matches with match errors below the match error threshold are inserted into ***F***. This process of creating matches continues, in the order of discernability, until all of the patches in the model have been searched for in the scene. At the end of the search, ***F*** will contain all full and partial feasible matches between the model and the scene.

The best rotation *R* for a combined match maps the axes of the model patches to the axes of the matched scene patches. It is found by minimizing the rotation error:

$$\sum_{i=1}^{N} \left\| R e_m^i - e_s^i \right\|^2 \tag{14}$$

$e_m{}^i$ are the model patch axes
$e_s{}^i$ are the scene patch axes

The best translation of the combined patch is found by manipulating Equation 7 and Equation 12 into the forms

$$A_m T = \frac{1}{2}(RB_s - B_m) \tag{15}$$

$$n_m^t T = d_s - d_m \tag{16}$$

respectively. Then for each match between quadric

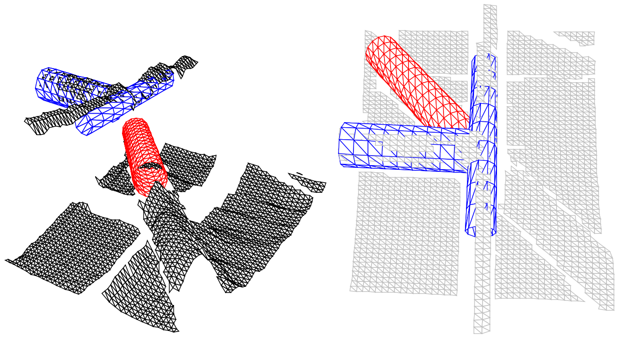

Figure 6: Oblique view (top) and sensor view (bottom) of the location of recognized objects superimposed on the surface mesh from Figure 4.

patches in the match, the three equations that satisfy Equation 15 are created, and for each match between planar patches in the match Equation 16 is created. These sets of equations are stacked to create one matrix equation that can be solved for T using singular value decomposition. SVD will produce the best solution in a least squares sense even when there are less than three axes matches which are required to fully determine the transformation.

The match error for a combined match is computed as the sum of the match errors as found from Equation 9 and Equation 13 for each match between patches in the combination.

Once all of the feasible matches have been found for the model in the scene, the best one is shown to the user in its calculated pose for verification. If the user believes that the model is in its correct pose, the model is accepted and it is inserted into the robots world model. This user verification eliminates the need for computer reasoning about feasible matches that would be required in a completely automatic object recognition system. The next best feasible match is presented to the user and the verification process is repeated until the user is satisfied with the modeling or the list of feasible matches is exhausted. This procedure is repeated for each model selected by the user. Figure 6 shows a located model pipe t-joint (made from two quadric surfaces) matched to scene patches T1 and T2, and a pipe (made from one quadric surface) matched to scene patch P superimposed on the mesh from Figure 4. The pose of each model comes from the best match of model to scene.

The combinatorial explosion of the search space is limited because after each model patch is searched for in the scene only the matches with a low error when combined with matches on the list of feasible matches are kept. Because the match error is low when patches are proximal in pose space, this is a form of a rigidity constraint. In addition to rigidity constraints, constraints on the surface area and curvature of patches that are matched are employed to limit the number of matches between a single model patch and the scene patches. The curvature constraint has been implemented by deciding which patches in the scene correspond to planes (by comparing the eigenvalues of the inertia matrix of the patch) and only matching planar model patches to planar scene patches and quadric model patches to quadric scene patches. The surface area constraint has been rudimentarily implemented by requiring that scene patches have a minimum surface area before being considered for matching

7 World Model

The final step in the modeling process is to present useful geometric information to the human operator. A virtual 3-D world provides richer understanding of the environment than 2-D camera data. To effectively convey a view of the robot in its workspace and the recognized objects a commercial robot simulation package, TELEGRIP from Deneb Robotics Inc., is used. This package allows viewing of the synthetic scene from any angle, manipulator path-planning, and off-line simulation of robot actions. The pre-stored model of the robot vehicle, manipulators and tooling is updated with the sensor-based models as they are constructed.A representative view of the robots world model is given in Figure 7.

8 Error modeling

The accuracy of the world model dictates the possible functions the operator can perform with the virtual world and the extent to which the sensed world geometry can be trusted as a basis for automatic motion and task planning. To perform a task, the operator must understand accuracy and errors that are present in the modeling process. Modeling the sources and propagation of errors through the system characterizes the errors present in the system and helps the operator to make decisions on what tasks are possible, given the current sensing conditions. Ultimately these errors depend on the error of individual points sensed in the scene and the errors in the control of the manipulator that performs the task. In the current scenario, the location of a point $P^o{}_r$ with respect to a coordinate system attached to a manipulator is specified based on sensor data. For example, $P^o{}_r$ is the location to which the manipulator must

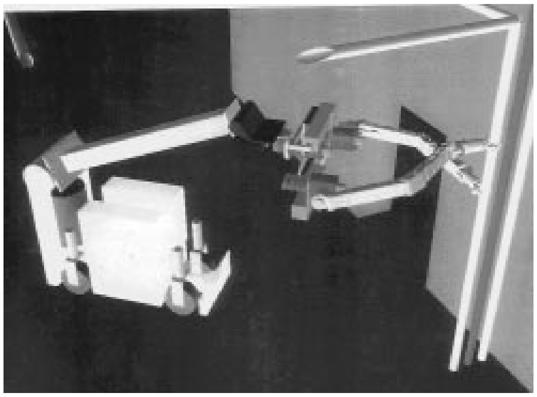

Figure 7: TELEGRIP model of robot workspace.

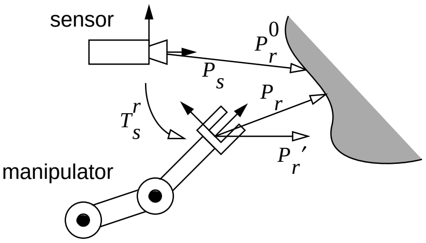

Figure 8: Geometry for error modeling.

move in order to cut a pipe. In this case $P^o{}_r$ would have been computed by fitting a surface to data points segmented out of a range image. Because of noise in the sensors, inaccuracies in the control of the manipulator and calibration errors, the manipulator is actually going to move to a different position $P'_r = P^o{}_r + \varepsilon$. The goal of this section is to characterize the error ε.

We characterize ε by its mean value E_f and its covariance matrix C_f ('f' stands for "final"). E_f characterizes the bias of the system. C_f characterizes the average variation of P'_r around $P^o{}_r + E_f$. Notice that, if all the errors in the system were unbiased, E_f would be zero. However, we will see that calibration errors do introduce a bias into the system.

8.1 Sensor errors

Assuming that the range sensor provides range measurements in spherical coordinates, $\rho(\varphi,\theta)$, the sensor error can be modeled by the standard deviation of errors on the measured range, σ_ρ, and on the scanning angles σ_φ and σ_θ. This model assumes that the measurement error has zero mean.

Typically, σ_ρ is a function of ρ^2, while σ_φ and σ_θ are constant [11]. With this model, the uncertainty C_s on the location of a point P of coordinates $P_s = [x_s\ y_s\ z_s]$, with respect to a coordinate frame attached to the sensor, corresponding to a measurement $\rho(\varphi,\theta)$ is given by:

$$C_s = J^t_s D_s J_s \qquad (17)$$

where J_s is the Jacobian of the transformation $S(\rho, \theta, \phi)$ from spherical coordinates to Cartesian coordinates. D_s is the uncertainty matrix of the measurement in spherical coordinates:

$$D_s = diag\left(\sigma^2{}_\rho, \sigma^2{}_\theta, \sigma^2_\varphi\right) \qquad (18)$$

Because the measurement errors are of zero mean, the error on the Cartesian point P_s is also of zero mean. Therefore, the covariance matrix C_s is sufficient to characterize the error. It is assumed that the range quantization noise is negligible compared to the measurement noise. This implies that the range is large compared to the resolution of the measurement. an assumption that is necessary for the covariance model to work. Other assumptions are that the mean range and angular error is zero and the errors are small enough that this linear approximation is acceptable.

8.2 Calibration errors

The coordinates of P are transformed to a coordinate system attached to the manipulator, $P_r = [x_r\ y_r\ z_r]$. The transformation between manipulator and robot coordinate system, $T^r{}_s$ is a fixed rigid transformation computed by calibrating off-line the manipulator with respect to the sensor. $T^r{}_s$ is composed of a rotation $R(\theta_x,\theta_y,\theta_z)$ and a translation $t = [t_x\ t_y\ t_z]$.

The error is a *fixed* error, as opposed to the sensor error which is *stochastic* in nature. More precisely, we can assume the calibration errors are represented by fixed errors $\delta\theta_{\{x,y,z\}}$ and $\delta t_{\{x,y,z\}}$ on the three angles of rotation and the three components of the translation, respectively. Denoting by $\delta T^r{}_s$ the transformation corresponding to the error parameters, the relation between the coordinates P_r in robot coordinates and the coordinates P_s in sensor coordinates can be written as:

$$P_r = T^r_s P_s + \delta T^r_s P_s \qquad (19)$$

Since $P^o_r = T^r_s P_s$, the error term in robot coordinates is $\delta T^r_s P_s$ which has mean E_r and covariance C_r given by:

$$E_r = \delta T^r_s P_s \qquad C_r = \left(\delta R^r_s\right)^t C_s \left(\delta R^r_s\right) \qquad (20)$$

In this expression, $\delta R_s{}^r$ is the rotation part of $\delta T_s{}^r$ (upper left 3x3 matrix). Assuming that the calibration errors are small, the matrix $\delta T^r{}_s$ can be easily calculated as:

$$\delta T^r_s = \begin{bmatrix} 0 & -\delta\theta_z & \delta\theta_y & \delta t_x \\ \delta\theta_z & 0 & -\delta\theta_x & \delta t_y \\ -\delta\theta_y & \delta\theta_x & 0 & \delta t_z \end{bmatrix} \qquad (21)$$

The assumptions inherent in this error modeling are that the errors are small enough for the linear approximation of $\delta T_s{}^r$ above to work and that the calibration of the sensor remains fixed

8.3 Manipulator errors

Assuming that a target point P_r has been selected from the sensor data, the manipulator is commanded to move to P_r. Errors are introduced in the control of the manipulator so that instead of reaching the ideal point P_r, the manipulator moves to a point $P'_r = P_r + \varepsilon$. Assuming position control, the error ε is due to the uncertainty on sensing the position of each joint q_i. More precisely, if each joint position θ_i (angle or linear displacement) is measured with a standard deviation σ_i, and if the kinematics of the manipulator is described by a relation of the type: $P_r = F(q_1,.,q_n)$, where Q is the vector of joint positions, the error ε on P_r is of zero mean and of covariance matrix C_t:

$$C_t = J^t_q C_q J_q \qquad C_q = diag(\sigma^2{}_1, ..., \sigma^2{}_n) \qquad (22)$$

In this model, J_q is the Jacobian of the kinematic transformation F.

The assumptions inherent in this error modeling of the manipulator are that the errors on joint positions are uncorrelated, the errors on joint positions are independent of the configuration, and the errors are unbiased.

8.4 Compounding of errors on a single point

In the model of the error attributable to the manipulator, the target point P_r is itself uncertain because it comes from a sensor measurement. Therefore, we need to compound (assuming that the errors introduced are uncorrelated) the errors due to sensor noise and to calibration. By combining the models developed above, we end up with $P'_r = P^o{}_r + \varepsilon_{sc} + \varepsilon$, where $P^o{}_r$ is the true location of the point in space, ε_{sc} is the error due to the combination of sensor noise and calibration error and ε is the error due to the manipulator.

Since ε is of zero mean, the mean error E_f on P'_r is the mean of ε_{sc} which was computed in the first section:

$$E_f = \delta T^r_s P_s = \delta T^r_s (S(\rho, \theta, \varphi)) \qquad (23)$$

where P_s is the point computed from the measurement (ρ,φ,θ).The covariance C_f of the error $\varepsilon_{sc} + \varepsilon$ is the sum of the covariances on ε_{sc} and ε:

$$C_f = \left(\delta R^r_s\right)^t \left(J^t_s D_s J_s\right)\left(\delta R^r_s\right) + J^t_q C_q J_q \qquad (24)$$

A complete error model for locating the position of a point in space has been presented. The error introduced on the position of the point after a surface has been fit to the sensed data has yet to be investigated.

9 Conclusion

Many internal details of the system operation have been described and illustrated, from the collection of the range image, through segmentation, to final model display. The system as used by a human operator hides most of this detail. The human operator initiates range image acquisition, chooses a region of interest, and selects the object models that exist in the scene. The planar and quadric segmentation and the subsequent location of object models in the scene are performed automatically. Located models that truly correspond to objects in the scene are then verified by the operator and inserted into the world model. The world model contains a model of the robot, all recognized objects and any available a priori information about the worksite, so it is an excellent tool for planning remote decontamination and decommissioning tasks.

In the near future, the complete system will be integrated to DOE's Dual Arm Work Module and then mounted on the Rosie worksystem and an overhead transporter. The mobility thus afforded will enable collection of multiple range images of the same task space. Future work will focus on means to effectively combine that data into more detailed, more accurate and more complete task space models.

Acknowledgments

This work is funded under contract DE-AC21-92MC29104 from the United States Department of Energy Morgantown Energy Technology Center. The authors wish to acknowledge the technical support and guidance of the DOE Robotics Technology Development Program, particularly Dr. Linton Yarbourgh and Dr. William Hamel, as well as the technical staff of the Telerobotics System Section of Oak Ridge National Laboratories and Sandia National Laboratories Intelligent systems and Robotics Center. We also acknowledge the programmatic support of Mr. Vijay Kothari of the Morgantown Energy Technology Center. Wenfan Shi, Herman Herman, Cecelia Shepherd and Marie Elm contributed technical support at Carnegie Mellon University.

References

[1] B.K. Christensen, et al. "Graphic Model Based Control of Robotics Systems for Waste Remediation." *Proceedings of the ANS Fifth Topical Meeting on Robotics and Remote Systems,* pp. 89-96, 1993.

[2] M. Trivedi and C. X. Chen. "Developing Sensor-based Robotics Systems Using Virtual Reality Concepts." *Proceedings of the ANS Fifth Topical Meeting on Robotics and Remote Systems,* pp. 165-172, 1993.

[3] A.J. Azarbayejani et al. "Recursive Estimation for CAD Model Recovery." *Proceedings of the Second CAD-Based Vision Workshop,* IEEE Computer Society, pp. 90-97, 1994.

[4] S. Thayer et al. "On-line Stereo Vision and Graphical Interface for Decontamination and Decommissioning Applications Using the Advanced Servo Manipulator." *Proceedings of the ANS Fifth Topical Meeting on Robotics and Remote Systems,* pp. 287-294, 1993.

[5] "Environmental Restoration and Waste Management Five-Year Plan Fiscal Years 1994-1995" Report # DOE/5-00097P, US Department of Energy, 1993.

[6] L. Conley, W. Hamel, and B. Thompson. "Rosie: A Mobile Worksystem for Decontamination and Dismantlement Operations." *Proceedings of the ANS Sixth Topical Meeting on Robotics and Remote Systems*, pp. 231-238, 1995.

[7] M. Hebert and E. Krotkov. "3-D Measurements from Imaging Laser Radars: How Good Are They?" *International Journal of Image and Vision Computing*, 10(3), pp. 170-178, 1992.

[8] O.D. Faugeras and M. Hebert. "The Representation, Recognition and Locating of 3-D Objects." *International Journal of Robotics Research*, 5 (3), pp. 27-52, 1986.

[9] M. Hebert, R. Hoffman, A. Johnson, and J. Osborn. "Sensor-Based Interior Modeling." *Proceedings of the ANS Sixth Topical Meeting on Robotics and Remote Systems*, pp. 731-737, 1995.

[10]P. J. Besl and N. D. McKay "A Method for Registration of 3-D Shapes." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14 (2), pp. 239-256, 1992.

[11]I. Kweon, R. Hoffman, and M. Hebert. "Experimental Characterization of the Perceptron Laser Rangefinder." Technical Report# CMU-RI-TR-91-1, The Robotics Institute, Carnegie Mellon University, 1991.